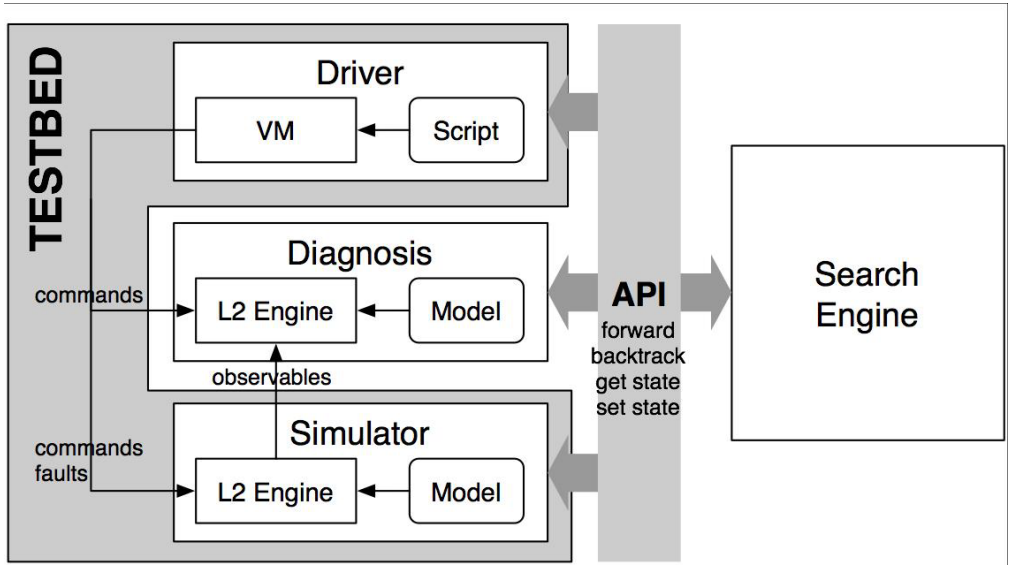
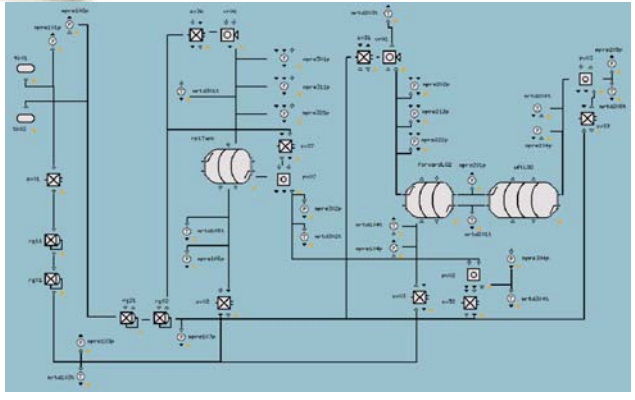
Simulation-Based Verification of Autonomous Controllers via Livingstone PathFinder





Goal: Verify that an autonomous controller performs correctly across a wide range of nominal and fault scenarios

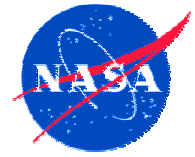
Approach:

- Apply optimized search algorithms to execute an instrumented testbed, consisting of the actual controller embedded in a simulated environment, using backtracking to explore alternate paths.
- Livingstone PathFinder tool (LPF) applies this principle to the Livingstone diagnosis system, searching for discrepancies between actual and diagnosed state.

Output:

- Successfully demonstrated on a large Livingstone model (X-34 propulsion feed subsystem).
- Paper presented:
  Tony Lindsey and Charles Pecheur. Simulation-Based Verification of Autonomous Controllers with Livingstone PathFinder. Tools And Algorithms For The Construction And Analysis Of Systems (TACAS'04), Barcelona, Spain, Mar-Apr 2004.
  See http://ase.arc.nasa.gov/pecheur/publi.html

Explanations



- **POC:** Charles Pecheur (RIACS, ARC / IC / ASE Group, pecheur@email.arc.nasa.gov) in collaboration with Tony Lindsey (QSS Group)
- **Program:** Engineering for Complex Systems (ECS) / Model-Based Hazard Analysis / Task item "Validation of Model-Based IVHM Systems"
- **Background:** Model-based autonomous systems pose hard verification challenges because of the huge space of unpredictable circumstances in which they are expected to operate. Livingstone 2 (L2) is a model-based diagnosis system developed at NASA Ames, intended for autonomy applications.
- **Accomplishment:** We developed Livingstone PathFinder (LPF), an advanced verification tool for Livingstone 2. The architecture of LPF consists of the L2 diagnosis system embedded in a testbed that comprises a simulator for the hardware (also based on L2) and a driver feeding commands and faults. This assembly is instrumented to allow precise control of its execution, using backtracking to explore alternate paths. LPF features a scripting language for specifying, in a flexible and concise way, a large set of execution scenarios to be analyzed. The search engine explores the tree of possible executions according to different strategies, searching for discrepancies between actual and diagnosed state.

  We have successfully applied that technology on a real-size model (X-34 propulsion feed subsystem, 800+ variables, bottom two pictures). This case study demonstrated the scalability of the approach to large applications, and also detected an error that lead to a minor adjustment of the X-34 model.

  This work was first described in a short workshop paper in June 2003, and in a full-size paper presented at TACAS'04, one of the prominent international conferences in Formal Methods, in April 2004.
- **Benefits:** Compared to conventional verification based on manually designed test cases, this solution provides a speedup by avoiding restarting the execution each time and to execute common initial sequences over and over again (for a uniform execution tree, this reduces the number of executed steps proportionally to the length of executions). As importantly, our tool also provides support for automating the enumeration of all executions, as well as for using heuristics to focus the analysis on selected cases, for example to maximize coverage criteria.
- **Future Plans:** We are currently completing the extension of the LPF architecture to Titan, a model-based controller from MIT. This will expand the application of simulation-based verification from Livingstone's purely passive diagnosis to Titan's reactive controller, including fault recovery. This is the next major objective of our project.